INSTITUTO FEDERAL
SANTA CATARINA

Indução e Recursão

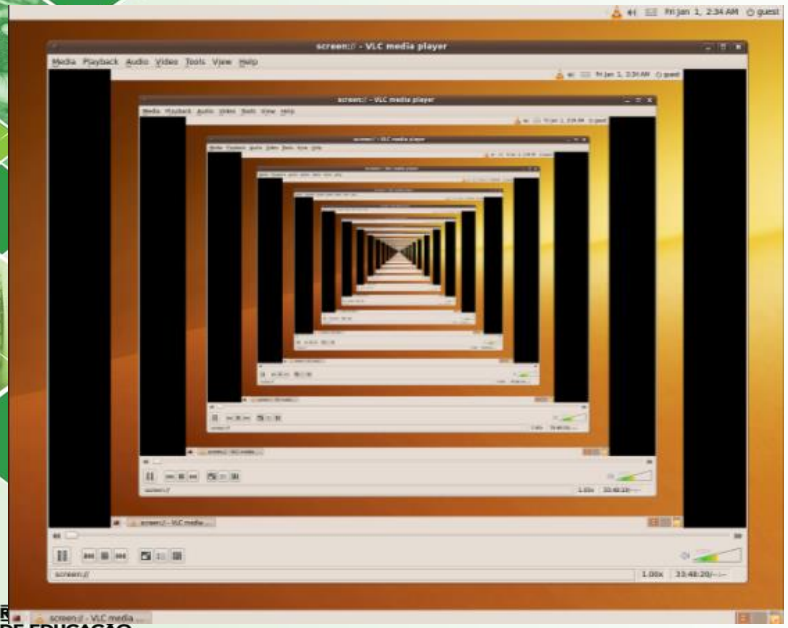
Matemática Discreta

105 ANOS REDE FEDERAL DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA 1909-2014

Prof° Marcelo Maraschin de Souza

Definições

Recursão ou recorrência: o objeto é definido em termos de si mesmo.



INSTITUTO FEDERAL
SANTA CATARINA

Definições

O conceito é usado:

- Na definição de sequências, funções e conjuntos;
- Na implementação de algoritmos.

Uma função *P* cujo domínio é o conjunto Z+ podem ser definidas recursivamente através de dois passos:

Passo 1) Especificar o valor de P(1);
Passo 2) Dar um regra (relação de recorrência) para encontrar seu valor para um inteiro a partir de seu valor para inteiros menores.

Tal definição é chamada de **definição recursiva** ou **definição indutiva**

Exercícios

Exemplo 1: na função P definida recursivamente abaixo, encontre P(2), P(3), P(4) e P(5).

$$\begin{cases} P(1) = 2 \\ P(n) = 2P(n-1), \quad para\ n > 1 \end{cases}$$

Exemplo 2: dê uma definição recursiva para $P(n) = a^n$**,** onde $a \in \mathbb{R}^*$ e $n \in \mathbb{Z}^*$.

Exemplo 3: dê uma definição recursiva para o somatório

$$\sum_{k=1}^{n} a_k$$

INSTITUTO FEDERAL
SANTA CATARINA

Exercícios

Exemplo 4: dê uma definição recursiva para $P(n) = n!, \text{com n} \geq 0$

105 ANOS REDE FEDERAL DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA 1909-2014

INSTITUTO FEDERAL
SANTA CATARINA

Algoritmos Recursivos

Definição (Algoritmo)

*Um **algoritmo** é um conjunto finito de instruções precisas para realizar uma operação computacional ou para resolver um problema.*

Definição (Algoritmo Recursivo)

*Um algoritmo é **recursivo** se resolver um problema reduzindo-o a um **mesmo problema** com valores iniciais menores.*

Algoritmos Recursivos

Tomando como base o exemplo 4, suponha que queremos escrever um código computacional para calcular P(n).

Com Recursão

```
1 int fatorial (int n){
2   if (n == 0)
3     return 1;
4   else
5     return n*fatorial(n
          -1);
6 }
```

Sem Recursão

```
1 int fatorial (int n){
2   if (n == 0)
3     return 1;
4   else {
5     int i, f = 1;
6     for (i=2; i <= n;i
          ++)
7       f = f * i;
8     return f;
9   }
10 }
```

INSTITUTO FEDERAL
SANTA CATARINA

Algoritmos Recursivos

Exercício: Tomando como base o exemplo 1, escreva um código computacional recursivo para calcular P(n). E desenhe um diagrama em árvore para n=4.

105 ANOS REDE FEDERAL DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA 1909-2014

INSTITUTO FEDERAL
SANTA CATARINA

Exercícios

1. Projete um algoritmo recursivo, em pseudo-código, que calcule o valor F_n da sequência de Fibonacci para o valor de n informado.

2. Desenhe um diagrama de árvore que ilustre as chamadas recursivas do algoritmo acima para $n = 6$.

Resolvendo Relações de Recursividade

Desenvolvemos um algoritmo recorrente para calcular o valor de P(n) no **exemplo 1**, no entanto, existe uma maneira mais fácil para calcular P(n)

Lembre que

$$\begin{cases} P(1) = 2 \\ P(n) = 2P(n-1), \quad para\ n > 1 \end{cases}$$

Então,

$$\begin{cases} P(1) = 2 = 2^1 \\ P(2) = 4 = 2^2 \\ P(3) = 8 = 2^3 \\ P(4) = 16 = 2^4 \end{cases}$$

Resolvendo Relações de Recursividade

Fazendo isso, podemos **conjecturar** que $P(n) = 2^n$.

Ou seja, podemos substituir n e calcular diretamente o valor de P(n).

Uma equação dessa forma é chamada de **solução fechada** para a relação de recorrência P(n) sujeita a P(1).

Quando pudermos encontrar uma solução fechada, dizemos que **resolvemos** a relação de recorrência.

Por fim, essa conjectura deve ser verificada por meio de **indução matemática.**

Resolvendo Relações de Recursividade

Uma das técnicas para se resolver uma relação de recorrência consiste na abordagem **“expandir, conjecturar e verificar”.**

Vamos mostrar o passo a passo para a recorrência

$$\begin{cases} S(1) = 2 \\ S(n) = 2S(n-1), \quad para\ n > 1 \end{cases}$$

Passo 1) Expandir a sequência S para n, (n-1), (n-2),...,k,...1

Resolvendo Relações de Recursividade

$$S(n) = 2 \cdot S(n-1) \qquad \text{(após 1 expansão)}$$

$$S(n) = 2 \cdot \left[2 \cdot S(n-2)\right] = 2^2 \cdot S(n-2) \qquad \text{(após 2 expansões)}$$

$$S(n) = 2^2 \cdot \left[2 \cdot S(n-3)\right] = 2^3 \cdot S(n-3) \qquad \text{(após 3 expansões)}$$

$$S(n) = 2^3 \cdot \left[2 \cdot S(n-4)\right] = 2^4 \cdot S(n-4) \qquad \text{(após 4 expansões)}$$

$$\vdots$$

$$S(n) = 2^k \cdot S(n-k) \qquad \text{(após } k \text{ expansões)}$$

$$\vdots$$

$$S(n) = 2^{n-1} \cdot S(n-(n-1)) \qquad \text{(após } n-1 \text{ expansões)}$$

$$S(n) = 2^{n-1} \cdot S(1) = 2^{n-1} \cdot 2$$

$$\boxed{S(n) = 2^n} \qquad \text{(solução em forma fechada)}$$

Resolvendo Relações de Recursividade

Passo 2) Após obter a solução fechada, **conjecturamos:**

Teorema (Conjectura)

$S(n) = 2^n$ *para todo inteiro* $n \geq 1$

Passo 3) Verificar a conjectura através de indução matemática para provar a forma fechada.

Resolvendo Relações de Recursividade

Prova por Indução.

- Passo Base: $S(1) = 2 = 2^1$
- Passo Indutivo: assumir que $P(k)$ é verdadeiro, isto é,

$$S(k) = 2^k$$

- Mostrar que se $S(k) \rightarrow S(k+1)$, i.e., $\boxed{S(k+1) = 2^{k+1}}$

$$S(k+1) = 2 \cdot S(k)$$
$$S(k+1) = 2 \cdot \underbrace{2^k}_{S(k)}$$
$$\boxed{S(k+1) = 2^{k+1}}$$

INSTITUTO FEDERAL
SANTA CATARINA

Resolvendo Relações de Recursividade

Exercício 1: encontre uma solução em forma fechada para a seguinte relação de recorrência:

$$\begin{cases} T(1) = 1 \\ T(n) = T(n-1) + 3, \qquad para\ n > 1 \end{cases}$$

Exercício 2: o jogo de “Torre de Hanói” tem a seguinte relação de recorrência

$$\begin{cases} T(1) = 1 \\ T(n) = 2 \cdot T(n-1) + 1, \qquad para\ n > 1 \end{cases}$$

Utilize o método “expandir, conjecturar e verificar” para encontrar uma solução fechada que dê o número de movimentações necessárias para completar o jogo com n discos.